Garbage Collection

Analysing terminology and metaphors in an uncooperative environment

Paola Karlsson
Abstract

This study analyses the translation of metaphors and terminology in an article regarding garbage disposal. The aim is to analyse problems occurring when translating metaphors and terminology in said article. While translating, terms were identified and translated with the help of a computer scientist and the results were analysed and discussed. Even though the study is not big enough to draw any major conclusions it was noticed that there was a difference in connotations found in texts discussing computer science and general meanings. What also was noticed was the usage of the coined Swedish terms in academic texts in contrast to the usage of borrowed terminology found on forums. Finally, the results showed that the terminology does not need to be translated to be understood since lectures and course literature are mainly written in English and the reason for using a Swedish equivalent is mainly to give the target text a cultural flavour.

Keywords: computer science, garbage collection, metaphors, terminology, translation
## Content

1. **Introduction** .................................................................................................................. 4  
   1.1 Aim .......................................................................................................................... 6  
2. **Material and Method** ..................................................................................................... 6  
   2.1 Material ..................................................................................................................... 6  
   2.2 Method ....................................................................................................................... 6  
3. **Background** .................................................................................................................. 7  
   3.1 Translation theory ....................................................................................................... 7  
   3.2 Terminology .............................................................................................................. 8  
   3.3 Metaphor theory ......................................................................................................... 9  
   3.4 Metaphors in Computer Science ................................................................................ 9  
4. **Analysis** ....................................................................................................................... 10  
   4.1 Terminology .............................................................................................................. 10  
   4.2 Metaphors ................................................................................................................ 16  
5. **Conclusion** .................................................................................................................. 20  
   Bibliography .................................................................................................................... 21  
   Primary Source .............................................................................................................. 21  
   Secondary Sources ......................................................................................................... 21  
   Parallel texts .................................................................................................................. 21
1. Introduction
Finding an up-to-date Swedish – English dictionary with accurate and used terminology within computer science is hard and almost impossible. The subject of computer science is very broad and it keeps getting bigger as new research is introduced to the world. It makes it hard to keep the terminology updated. Since most research is carried out and presented in English as a lingua franca it is even harder for people with another mother tongue to find an accurate translation for a certain word. This leads people who work with text based on computer science to borrow words from the English language and use their own grammar to mould the word into something fitting for a text, in a new language. Doing this is an easier way to recognize and connect the proper terminology between languages, but problems might occur when a term has a formal translation, or if a term is on its way to become an accurate synonym of the English word. It leads the translator to question which word s/he should use while adapting the text into the new language. As it has been done throughout this thesis, and many translators of computer science texts will do, googling terminology is the easiest way of finding out if a term is more used in certain areas of this field.

What should be noted in these kinds of texts is the usage of metaphors; how they are used as term is and how they accurately describe events in the user’s computer. While translating the text one can see that the metaphors found in the Swedish text are very similar to each other as they, as well, have been borrowed, and/or, translated.

In this paper I look at how it is possible to translate terminology from English to Swedish in a text about Garbage Collection, i.e. by looking at the difficulties of translating a text where the connotation of a word normally does not relate to the terminology used in a computer scientific text. The example below illustrates the problems encountered in the translation of terminology:

(1) Conventional automatic storage       Konventionell automatisk
[p 1] management systems rely on the   lagringshantering förlitar sig på
user program or (…)  programmet, (…)

As one can see in (1) above, the words *user program* is translated into *programmet*. Translating said term into *användarprogrammet* would not agree with parallel texts since the word does not refer to a program used by a human being but to a program which uses the garbage collector. Making a simple translation mistake, like in this case calling *Garbage collector* to *Sophanterare* or *Sopsamlare*, will make the translator very visible in the target text and will furthermore show his or her lack of understanding of it.

Furthermore, I will look closely at the metaphors in the source text (*henceforth ST*) text and see what kinds of translation methods are used for the terminology; find a way into the Swedish text with only minor adaptations? Looking at the next example will show how a metaphor is borrowed from the English language and translated into an established Swedish equivalent:

(2) In any practical programming language implementation, compatibility with the underlying operating system and existing *program libraries* is an issue

*garvage collection in an uncooperative environment* was written by computer scientists for other computer scientists, and people who already are familiar with one set of terminology will be confused if the translator decides to use an unfamiliar set of terminology, which is why it is important produce an accurate translation.
1.1 Aim
The aim of this paper is to analyse problems occurring when translating metaphors and terminology in an article about computer science.

2. Material and Method

2.1. Material
This paper will be focused on analysing my own translation of approximately 10 pages of the article *Garbage Collection in an Uncooperative Environment*. The article was written in 1988 and the audience is a very close-knitted group of people with a vast amount of knowledge in the field of garbage collection. It describes a way of using storage allocation and garbage collection as well as simplifying languages which support said garbage collection. The ST was selected due to the advanced nature of a text regarding Garbage Collection and a way for me to develop my skills regarding computer scientific terminology.

2.2. Method
By looking at the differences in terminology and how a specific term should be used in this text I will show how the end results were achieved. Moreover, I will look at the usage of metaphors in a text on computer science since the language used in it is mainly based on metaphors.

Finding the terminology itself in Swedish and figuring out which of the synonyms to use was difficult since one had to guess what the Swedish terminology could be. Since computer science texts use direct loans and loan translations it was difficult to find out which terms should be used in the subject of garbage collection. This was done through consultation with Erik Österlund, who studied computer science at Linnaeus University while working at Softwerk as a programmer. Essays from other universities as well as examples from their course literature were used as parallel texts to help me support my choice of translation, or lack thereof. Google, as well as my own knowledge in the relevant field will also be used throughout my analysis.
3. Background
This section presents the theories that will serve as a basis for the analysis of the translation.

3.1. Translation theory
There are different ways to translate a text; in Vinay & Darbelnet’s *Comparative Stylistics of French and English* (1995) the authors show different methods that can be used as tools for translating a text. The ones we mainly use, according to them, are literal translations or oblique translations (1995:31). They argue that the most common use of literal translations is when translating between two languages of the same family (1995:34) which in this case would be Swedish and English, and also where the two languages might share the same culture. A literal translation from English to Swedish could be *cast iron* translated into *gjutjärn*.

However, if the source language (henceforth SL) and the target language (henceforth TL) belong to different language trees, it might be hard to find a literal translation since the translation should be reversible and complete in itself without any modification (1995:34).

If a translator, according to Vinay & Darbelnet (1995:34), finds a literal translation impossible they should move on to an oblique translation. They define an impossible translation as one where the translation would fulfil any of the following criteria:

1. It gives another meaning than what the SL describes or,

2. has no meaning or,

3. be structurally confusing or,

4. not have the same expression in the TL or,

5. have an expression but not the exact one (1995:34–45).

While Vinay & Darbelnet mostly focus on the aspects of translating words and phrases, Newmark in his *A Textbook of Translation* (1988), adds to the discussion with his own theories on how texts may be translated, and discussed since there is no set rule for a translation (1988:20).
He describes the two approaches to translation, i.e. translating sentence by sentence, reviewing and editing or reading the SL text, analysing the text in order to understand it (1988:21). However, like Vinay & Darbelnet, he emphasises that a text has to be natural (1988:34). For the text to have a natural flow, the translator has to stay invisible in his/her translation and also try to translate the effect the SL has on the reader (1988:48). Newmark also remarks the naturalness of a translation, writing that the text should be fluent and coherent for the readers in the TL. Furthermore, Newmark states that it is important for translators to work with a TL which is of habitual usage since they have to detach themselves from the SL (1988:26). This leads us into the next subsection of the background which is metaphors. For a text to be natural, the usage of metaphors has to have an equivalent and be coherent in the TL.

3.2. Terminology
Rune Ingo (2007:83) describes language for specific purposes (LSP) as a language used for specialised knowledge within a range of fields such as law, administration and technology. LSP is, according to Dita Gálová (2007:10), focused on [the] learners’ professional, linguistic needs. Furthermore, Gálová (2007:10) states that the origin of LSP is closely connected to recognizing the communicative role of languages and their functional styles, meaning that LSP is used as a sub-language, essential for specialised communication.

In “Terminology: theory, methods and applications”, Maria Teresa Cabré (1999:3-4) mentions a paradoxical struggle between monolingualism and the usage of a national language due to the fact that national languages are recognised as tools of communication, both generalised and as an LSP whereas monolingualism is mainly needed for a direct and efficient communication. Cabré reasons that there is a need for a language standardisation where the language should be allowed its own cultural aspect but at the same time be a part of something greater, such as a national language. A combination between these two is sometimes found in technological translations which will be seen in the analysis in Section 2.1.

Furthermore, Cabré also points out that technology is growing and imbues society which is why there is a need to create new ways of communication which constantly need to be updated as technology changes. Moreover, Cabré (1999:161) states that terminology, specifically in computer science, is the basis of all knowledge and is needed for said field to advance.
3.3. Metaphor theory
Metaphors are used in everyday language and are something we cannot avoid since they are something we are not aware of, according to Lakoff & Johnson (1980:1). For this essay I will use the definition used in “Metaphor in Computer Science”, by Colburn & Shute where they define a metaphor as “a description of an object or event, real or imagined, using concepts that cannot be applied to the object or event in a conventional way” (2008:1).

Metaphors have some sort of cultural value which is why one can visualise the context of the metaphor and not try to define it in a literal sense. Lakoff & Johnson use the metaphors “Time is money”, “I’ve invested a lot of time in her” and “Thank you for your time” to show that time is of cultural value in Western culture (1980:8). According to the authors, we can relate to the phrase “Time is money” due to our culture, i.e. paying bills, wages and debts (1980:8).

3.4. Metaphors in Computer Science
In the article “Metaphor in Computer Science”, Colburn & Shute show that “the language of computer science is laced with metaphor” (2008:1) which they show by defining some metaphors used in a text:

(3) Programmers needing to effect an exit from a procedure that involves a complicated transfer of processor control can *throw* an exception while needing to know nothing about how such an action affects the runtime stack.

Colburn & Shute indentify different kinds of metaphors and divide them into two groups, comparative theories and interaction theories. They also write that metaphors in a computer science text cannot be defined in the same way since some of the metaphors do have a relation between the source and target in a physical way (2008:2) while others do not. Metaphors in computer science are made by both using already existing metaphors and also by making their own with the help of the world they are creating such as the metaphor for the icon *desktop recycling bin*. (2008:4-5).

Metaphors are also used pedagogically in computer science. Colburn & Shute use the “Add to Shopping Cart” metaphor to show that a customer will only need to click one button to understand the great amount of information given to them since there is a physical relation
between clicking the icon of a shopping cart and actually going shopping in a physical store (2008:4). However, they are also used as explanatory tools in computer science. When problems occur in one’s computer or data, there has to be an answer to what is happening and Colburn & Shute use a flow metaphor to show how it would work (2008:6). Flow can be imagined as water flowing through a pipe (2008:6) which is what the metaphor wants to make the user visualise while reading about the flow in a computer. The same thing might be seen in queue, where one can visualise a queue just as several tasks in one’s computer stand in line to be performed (2008:6).

4. Analysis

4.1. Terminology

The most important term to translate in an article about garbage collection is the actual term garbage collection. Since the term itself is not only a part of the terminology but also a metaphor which will be analyzed further on I will briefly analyse my reasons for using skräpsamling. Compounds are frequent throughout this text and skräpsamling and skräpsamlare fall under this category and are thus analyzed together.

The term garbage collection can be translated into sophertering, meaning the exact same thing as skräpsamling, but for computer scientists it will have a different connotation. While googling sophertering + datavetenskap I found no hits which were related to the garbage collection referred to in this essay but about actual garbage collection. The reason for trying the term sophertering was my direct association with handling garbage, which is what a garbage collector does in a computer. On the other hand, the search words skräpsamling + datavetenskap gave me 105 hits, most of which were from departments of computer science at different universities throughout Sweden who use skräpsamling as their translation of garbage collection. The Royal Institute of Technology in Stockholm uses the term skräpsamling (2007:70), for example in a study manual from 2007–2008. Nationalencyklopedin [www] confirmed my assumptions as well with its definition: “garbage collection (engelska, ’skräpsamling’), i databehandlingssammanhang en process vid dynamisk minnesanvändning där tidigare utnyttjade minnesceller…”;

Skräpsamling can therefore be seen as the literal translation of garbage collection. With help of
some examples found throughout the translated article, I will analyise how conclusions were drawn with regards to the translation.

What will be analyzed separately is the word skräpsamlingstider. The reason is that skräpsamlingstider was such an uncommon term and it was only found in one parallel text.

(4) In general, garbage collection times are likely to remain reasonable, even for interactive applications, provided the total amount of memory in a particular address space is not excessive.

Fortunately, this was in “Optimizing Memory Performance with a JVM: Prefetching in a Mark-and-Sweep Garbage Collector” (Hallberg 2003) a Master of Science thesis from Department of Microelectronics and Information Technology on Royal Institute of Technology in Stockholm where a student in her Swedish summary writes: “De experiment som utförts visar en tydlig förbättring av skräpsamlingstiderna med den inlagda prefetchoptimeringen”. Even though one source should not be enough for a term to be recognised, this might be a new term, coined by Hallberg and approved by the Royal Institute of Technology. Since they have an established department of Computer Science, one could assume that this is the origin of a new term which might be used in future research and articles alike. A coined term in computer science will most likely be used if it has been printed in an article and well as defined by the author. Seeing this as the rise of a new term I decided to use it in my TT. Moreover, using the English term garbage collection times in this case while using the Swedish translation skräpsamling in others, could confuse the reader and make the text less fluent.
The case of using borrowing words in a computer science text is quite common since most of the terminology in the TT is English.

(5) we modified the marking procedure to ändrade vi märkningsproceduren så den
[p 9] report ‘near misses’... rapporterade ”near misses”...

This occurs mostly because many computer scientific texts and the course literature used at universities are not translated into Swedish and it is therefore easier for the reader to understand the English term better than a coined one. Moreover, the courses given at many universities are taught in English, which could also confuse the student if s/he had to keep translating terminology.

By looking at courses in computer science given at Linnaeus University one can see that this is true. The course *Utvalda problem inom kompilatorkonstruktion, 7.5* [www] is given in English, and by looking closer at the course literature, one can see that all the material found is in English as well. Books such as *Compilers - Principles, Techniques, and Tools* by Aho, Lam, Sethi, & Ullman, and *Advanced compiler design and implementation* by Kaufmann are found in the list of required literature, and both are in English.

In (6) below one can see that the English term is used, and even though the sentence combines Swedish and English it is understandable for a computer scientist according to Erik Österlund.

(6) In the general case, this forces Allmänt, så tvingar det activation records
[p 1] *activation records* and/or *closures*, as och/eller *closures* såväl som variable celler att
well as variable cells to be heap bli heap-allokerade
allocated

Moreover, by googling the term *activation records* on “pages written in Swedish” (Sidor skrivna på svenska), a feature found on Google, the terminology is in English but with the Swedish term
in parenthesis (aktiveringspost), or in English as found on Royal Institute of Technology’s course page for Kompilatorer och exekveringsmiljöer 7,5hp[www]. I find the term activation records to be a term which has a Swedish equivalent (aktiveringspost) but it is not used to the same extent as to describe this kind of activation record but the one which is also called “frame” which can be seen in the lecture notes of Introduktion till datalogi at the Royal Institute of Technology:

“Dessutom kommer det så länge metoden main exekverar (dvs hela tiden) finns en aktiveringspost (frame) för mains parameter iargs samt mains lokala variabler, b och c.”.

This shows that activations records have more than one meaning in computer science.

The same borrowing is seen in several cases in the text where one either borrows the word in its total or when the words are borrowed and determined by applying Swedish grammar to the English word which is seen in (7).

(7) A slight variation on the same garbage collector was supplied to students in support of a compiler class project requiring storage management in the run-time system.

En variant av samma skräpsamlare gavs som stöd till studenter som gjorde en gruppuppgift med en kompilator som behövde lagringshantering i runtime-systemet.

Furthermore, by looking at the examples below, where the terms are adapted into Swedish, it is seen that the most common way of handling the computer scientific terminology in this text is to borrow the set terminology, as in (8), and in some cases use the term with another Swedish term or turn them into compounds such as in (9) and (10).

(8) Our approach relies on the use of a mark-sweep collector.

Vårt tillvägagångssätt bygger på användningen av en mark-sweep samlare.

(9) (See Reference [13] for a survey of garbage collection algorithms.)

(Se referens [13] för en undersökning av skräpsamlingsalgoritmer.)

(10) typically interface routines must be provided to perform the appropriate translation

Vanligtvis måste interface-rutiner tillhandahållas för att utföra en lämplig översättning.
The word *heap* in itself can be translated into *stack*, which is another term used in computer science. Although the translation is semantically precise there is almost no relation between the word *heap* and the word *stack*. A brief explanation of both will clarify the matter.

A *stack* in computer science is a linear data structure meaning that the user can only withdraw something from the stack which he/she has recently placed there and everything else in the stack is hidden. In a *heap* on the other hand, all data is available at all times which makes it more complicated. Both the stack and the heap are used in the memory but they work in different areas.

This leads me back to my assumption that if a translator does not know the difference between these words and relies on Google to find the result with most hits, one will see that *stacken* has more results when googling *stacken + datavetenskap* than *heappen + datavetenskap* which also means that the translator has chosen an incorrect term for his or her translation. To verify that the term “heappen” was used in academic texts as well, parallel texts containing closely related subjects were found. Karl-Johan Karlsson, uses the word heappen in his essay ”Generationsskräpsamling med explicitkontroll av hårdvarucache” ”En ytterligare förbättring skulle kunna vara att lägga strängtabellen i Scheme-heappen, eller lägga strängar direkt i Scheme-heappen utan att använda en strängtabell alls.”

To see that it was not only Karlsson who uses this term, further research was done and the term was also found in papers handed out by the faculty of engineering at Lund’s University. Here, the sentence ”efter borttagningen har heappen det utseende som visas till höger i figuren nedan” was found. With that in mind, the conclusion was drawn to use the term *heappen*.

Similarly, we may divide the heap into different regions, such that all the objects in each region have the same type and thus an identical storage layout.

På liknande sätt kan vi dela upp heappen i olika regioner, så att alla objekt i vardera region har samma typ och därmed en identisk minneslayout.
Furthermore, some terminology has no real translation in Swedish nor is the English terminology borrowed which in some cases can make translation hard. The term *allocation-and-free* is particularly relevant. Searching for *allocation-and-free + datavetenskap/skräpsamling* and *allocation and free + datavetenskap/skräpsamling* was futile since there were no hits at all. Consulting with Erik Österlund showed me that there was a direct translation for the word which meant that I had to find an equivalent translation and re-write my sentence, adapt it to the Swedish term, which in this case was *allokerar och frigör minne*. Since terminology has a specific meaning to it, I had to figure out the meaning of an allocation-and-free tracer before trying to write my translation. As seen in the *heap/stack* example, it is necessary to find the correct connotation of the words used to describe the term. In (12) I therefore opted for a more free translation which does not hold a resemblance to the ST.

(12) The tool is an *allocation-and-free* tracer, which is invoked optionally as part of the collector at run-time. It works as follows.

Verktyget är en *sökare som allokerar och frigör minne*, och anropas valfritt som en del av samlarens run-time.

Loan translations, literal translations, are also fairly frequent but seemingly the translators or authors can sometimes choose whether they want to use the Swedish or English term if both are equally accepted. In the case of *C compiler* and *C-kompilator* both are used, although one is more frequent on Internet forums (C compiler) while the other (C-kompilator) is used on official University homepages and forums according to a quick Google search on *c-kompilator* and *c compiler*. The term *c-compilern* is also found online but mainly on forums where the usage of mixed language and terminology occurs more frequently than in papers, written at universities throughout Sweden.

In contrast to the former example, *allocator* and *allokator* are terms which are set depending on the language used. The Swedish term *allokator* is a set term according to computer scientists. Not even the usage of the English *allocator* was found on Google while searching for *allocator + datavetenskap* on pages written in Swedish while *allokator + datavetenskap* and just *allokator* gave a greater number, around 11.200, of results, regarding of computer science.
It can be used with code produced by, say, a conventional C compiler. Den kan användas med kod som producerats av, till exempel, en konventionell C kompilator.

The storage allocator assures that given such a data value, we can determine whether it points to a valid object that is administered by the allocator. Lagringens alloktator garanterar att om den skulle få ett sådant datavärde, kan vi avgöra om den pekar på ett giltigt objekt som är administrerat av allokatern.

Literal translations were said to be the most common translations according to Vinay & Darbelnet, which can be seen throughout my examples. Even though words have been compounded, such as *skräpsamling*, they are still direct translations from the ST and can be used in a correct manner throughout the TT. The similarities between the two languages make it easier to keep a literal translation throughout the TT. In the next section there will be a discussion on how some, metaphors can be used as terms too, which will be further analysed in examples such as the term *garbage collection*.

### 4.2 Metaphors
Colburn & Shute (2008:1) write that metaphors are widely used in texts such as the ST, since they serve more than one purpose. In the TT, the metaphors mostly serve as part of the terminology and as verbs, showing what the object will, or can do. The metaphors in themselves will show how, like in the matter of *heap/stack*, a metaphor can be confusing to the reader if not translated properly.

Like in section 4.1, examples will be used to describe different kinds of metaphors found throughout the translated text.

Starting off with a metaphor, which is also a term, *garbage collection* is a metaphor which describes an item. The metaphors *garbage* and *collection / skräpsamling* are meant to describe a collector, which does not physically exist, who tries to recover memory no longer used by certain programs in your computer. In this example one can see the literal translation of the metaphor...
from English to Swedish and consequently it is easy for the user to associate both versions with each other. In this case the different ways of translating the terminology has previously been discussed.

A literal translation of a different kind is also found in (16) where program libraries have been translated into programbibliotek. This serves a purpose for the reader who can visualise a library where it does not exist.

(16) In any practical programming language implementation, compatibility with the underlying operating system and existing program libraries is an issue. I all implementering av programmeringsspråk är kompatibiliteten med det underliggande operativsystemet och existerande programbibliotek ett problem.

In contrast to a physical library, the equivalent in a computer is not built or organised in the same way. Phrases such as example(16) is not only found in advanced texts regarding this matter but also in music programmes, like iTunes, where the user can add certain objects to a specific place on the computer for their faster usage. The Apple support group defines one of these libraries as such “When you add something to the library, essentially you are taking existing audio or movie files and adding an entry for each to your iTunes Library index” [www].

This is very similar to Colburn & Shute’s (2008:4) explanation of pedagogical metaphors found in computer science. Colburn & Shute state that users of new software may be like children when they use the tools in the software and they acquire language in the same way which means that painting up pictures for the inexperienced would be the easiest way for that person to learn about a new subject.

Moreover, Colburn & Shute discuss Piaget’s way of describing children’s way of projecting which is also applicable to any person using a computer:

Examples from the study of children include situations of pretending, as when a child plays with a dollhouse by projecting her schema of ordinary life in a home onto her dolls and miniature furniture. When a user of web site software first encounters a web shopping cart, she must project her schemas involving ordinary shopping carts onto the web shopping cart (2008:4).
In (15), (16) as well as in (17), the user has to project, or visualise, the word mentioned. A routine in a computer is not tailored by a tailor but rather designed so it can be used by the program.

*Skräddarsydd* is a literal translation of *tailored*, and both words hold the same connotation in English and in Swedish. The English corpus, British National Corpus [www], shows examples of *tailored* referring to, for example, clothing, “Lucy had style, from well-cut red-gold hair down through the subtly *tailored* suit,”, but also how something could be tailored to fit a person’s needs: “who could coordinate an educational programme specially *tailored* to each senior house officer's needs”. The Swedish corpus, Korp [www], also shows different examples of *tailored*: En lättviktsbyxa med *skräddarsydd* passform” and lastly, as a figurative meaning: Politiken måste därför vara *skräddarsydd* för den lokala eller regionala arbetsmarknaden. Both corpora had around 500 hits each and the examples were taken from early 1990’s to 2012.

Furthermore, the same strategy can be applied to (17), where a memory “leak” does not refer to an actual leak of memories from a code, which would physically be impossible but to our mental images when we think of a leak.

Colburn & Shute (2008:4) also state that this is not the only way a child or a user will learn how to use metaphors. While just projecting the image might give the user a sense of meaning, s/he must be able to learn and grasp the concept of the metaphor.

(16) In the presence of static typing information in the source program, it is possible to generate a traversal routine specifically *tailored* to a user program [8].

(17) … but is perhaps even more useful when the code being debugged was written without a collector in mind, and so should be free of *memory ‘leaks’*.  

Furthermore, the same strategy can be applied to (17), where a memory “leak” does not refer to an actual leak of memories from a code, which would physically be impossible but to our mental images when we think of a leak.

Colburn & Shute (2008:4) also state that this is not the only way a child or a user will learn how to use metaphors. While just projecting the image might give the user a sense of meaning, s/he must be able to learn and grasp the concept of the metaphor.

(16) In the presence of static typing information in the source program, it is possible to generate a traversal routine specifically *tailored* to a user program [8].

(17) … but is perhaps even more useful when the code being debugged was written without a collector in mind, and so should be free of *memory ‘leaks’*.  

Furthermore, the same strategy can be applied to (17), where a memory “leak” does not refer to an actual leak of memories from a code, which would physically be impossible but to our mental images when we think of a leak.

Colburn & Shute (2008:4) also state that this is not the only way a child or a user will learn how to use metaphors. While just projecting the image might give the user a sense of meaning, s/he must be able to learn and grasp the concept of the metaphor.

(16) In the presence of static typing information in the source program, it is possible to generate a traversal routine specifically *tailored* to a user program [8].

(17) … but is perhaps even more useful when the code being debugged was written without a collector in mind, and so should be free of *memory ‘leaks’*.  

Furthermore, the same strategy can be applied to (17), where a memory “leak” does not refer to an actual leak of memories from a code, which would physically be impossible but to our mental images when we think of a leak.

Colburn & Shute (2008:4) also state that this is not the only way a child or a user will learn how to use metaphors. While just projecting the image might give the user a sense of meaning, s/he must be able to learn and grasp the concept of the metaphor.
Some metaphors are easier than others to relate to, even if they are used differently, such as *skräddarsydd*. However, *minnesläckor* is a term which needs to be acquired to be understood. A leak does not refer to something actually leaking out of a memory, but a program using up too much memory. The term *memory leak*, and the Swedish equivalent *minnesläckor*, are terms which are tightly bound to their specific domains and cannot therefore be easily understood by people without said knowledge. Another example of a metaphor which needs to be acquired is the *conservative garbage collector*.

(18) "Conservative” Garbage Collectors "Konservativa” skräpsamlare

While according to the Cambridge online dictionary [www], *conservative* refers to something “…often not liking or trusting change, especially sudden change”, one can associate conservative in many different ways. The conservative garbage collector has a different meaning. Erik Österlund explains that in the memory, there are memory-cells with references, linking from one to another. A conservative garbage collector does not have to know exactly what these references are, in contrast to an accurate garbage collector where the collector needs to know precisely where all the references are located and what they are. What also is seen in this brief explanation of a *conservative* garbage collector is that the antonym does not relate to what is normally seen as the opposite of conservative.
5. Conclusion

In this thesis, the difficulties in translating terminology and metaphors in computer science were studied. The terminology analysed were both general technical terms within said field but also terms only found in texts regarding garbage disposal.

The ST was written by Hans-Juergen Boehm, to a very small target audience with a vast knowledge of the area of research. The background was provided by studying translation theories by Vinay & Darbelnet and Newmark, terminology by Ingo, Gálová and Cabré and metaphors by Lakoff & Johnson and Colburn & Shute. With help of parallel texts touching on the subject of garbage collection, but also some with a more general knowledge of computer science an analysis was made. The method for analysing and finding the term used by computer scientists was very similar in all cases. With the help of a computer specialist, some terms were given to me, verified with the help of dictionaries, parallel texts and Google searches, while others had to be directly translated and with the help of Google searches I had to find the translation which was used by the rest of the computer scientific society.

Texts regarding computer science are interesting to translate, although one can argue that a translation might not be needed since the literature used in many courses regarding computer science are written in English, and the language used during these classes is English as well. What is also mentioned and seen on the Internet is the frequent usage of English terms, such as C compiler, on Swedish forums while the coined translation, C-kompilator, is used in academic texts and articles.

Moreover, what has been noted throughout the translated article is that many metaphors found, are used in daily conversation but their connotations differs due to how the language in computer science is built. While looking at the text and comparing the results to Colburn & Shute one can see that they paint up a similar picture in both languages and the pedagogical method applied throughout the text and others help the learner, or student in the usage of computers. Future research could include code-switching in computer scientific articles as well as assembling a term database which would be officially used in Swedish.
Bibliography

Primary Source
Boehm, Hans-Juergen. 1988, Garbage Collection in an Uncooperative Environment, Computer Science Department, Rice University, Houston.

Secondary Sources
British National Corpus. http://www.natcorp.ox.ac.uk/ (8 June 2012)


Cambridge Dictionaries Online http://dictionary.cambridge.org (8 June 2012)

Colburn, T.R. & Shute, G.M. 2008, Metaphor in computer science. Department of Computer Science, University of Minnesota, Duluth, MN.


Språkbanksens korpus, Korp. http://sprakaanken.gu.se/korp (8 June 2012)


Parallel texts


Kompilatorer och exekveringsmiljöer 7,5 hp ( http://www.kth.se/student/kurser/kurs/ID3006).

(8 June 2012)

KTH Studiehandbok 2007-2008,  
http://www.kth.se/polopoly_fs/1.106569!/Menu/general/column-content/attachment/Studiehandbok_del%204_200708.pdf (8 June 2012)

Metoder som returnerar värden http://www.nada.kth.se/kurser/kth/2D1340/nda01/f7.html (8 June 2012)

Utvalda problem inom kompilatorkonstruktion, 7.5 (http://lnu.se/utbildning/kurser/4dv008)

(8 June 2012)